WIKITAAABLE: A semantic wiki as a blackboard for a textual case-based reasoning system

Amélie Cordier[1], Jean Lieber[2], Pascal Molli[2],
Emmanuel Nauer[2], Hala Skaf-Molli[2], Yannick Toussaint[2]

[1] Université de Lyon, CNRS,
Université Lyon 1, LIRIS, UMR5205, F-69622, France.
{firstname.lastname}@liris.cnrs.fr
[2] INRIA Nancy -Grand Est
Nancy Université
LORIA – UMR 7503. B.P. 239, F-54506 Vandœuvre-lès-Nancy cedex, France.
{firstname.lastname}@loria.fr

Abstract. Semantic wikis enable a community of users to produce formalized knowledge readable and usable by machines. To take one step further, one can use a semantic wiki as a blackboard allowing humans and machines to interact in order to build knowledge that is useful for both humans and machines. In this paper, we present a case study of the use of a semantic wiki (Semantic Media Wiki) as a blackboard to manage culinary data and knowledge. This case study is performed within the context of the TAAABLE application, a case-based reasoning web system aiming at solving cooking problems on the basis of existing recipes. With WIKITAAABLE, an evolution of TAAABLE based on a semantic wiki, we show how a semantic wiki assists users in their knowledge management tasks by taking into account user feedback. The issues related to the integration of several knowledge management mechanisms in a single application are discussed at the end of the paper.

1 Introduction

Wikis have demonstrated how it is possible to transform a community of strangers into a community of collaborators. By integrating Semantic Web technologies, semantic wikis [1–3] allow this new community of contributors to produce formalized knowledge readable by machines. Maybe the next step is to use semantic wikis as a blackboard where humans and machines can interact together for producing knowledge that is updatable by humans and machines.

We investigate in this paper how this can be achieved using the TAAABLE system [4] as a case study. The TAAABLE system[3] allows users to query a cooking recipe base to solve cooking problems. For example, a user can submit the following request: "I want a dessert with rhubarb but without chocolate". If no recipe exists with the specified characteristics, an existing recipe is adapted

[3] The TAAABLE system is accessible online at http://taaable.fr/

in order to answer the request. The system relies on a case-based reasoning (CBR [5]) engine to perform adaptation. The CBR engine uses different data and knowledge sources: a set of indexed recipes, an ontology of ingredients, types of dishes, geographical origins of dishes and types of diets (vegetarian, nut-free, no alcohol) and a set of adaptation rules. The indexed recipes are computed from recipes written in natural language. An indexing tool uses the different ontologies to index the recipes.

If the system is working fine, the maintenance of recipe base and knowledge is cumbersome. The indexing tool performs natural language processing which is error prone. In addition, the system is not able to capture the users feedback to improve its internal adaptation capabilities.

In this paper, we study how a semantic wiki can be helpful in the TAAABLE system for managing data (e.g. cooking recipes, terminological information in the domain of cooking) and knowledge (e.g. ontology and adaptation rules) feeding the CBR engine.The semantic wiki is used as a blackboard where humans and computers interact to produce knowledge. Humans can add new recipes, correct indexing of existing ones and give feedback about the results of adaptation. Machines can perform adaptation of users' queries and indexing of recipes. Machines can also take into account user feedback in order to improve adaptation and indexing.

The paper is organized as follows: section 2 describes the current application. Section 3 describes how we are building the WIKITAAABLE system based on Semantic Media Wiki. The last section concludes the paper and points out future works.

2 The TAAABLE project

The TAAABLE project was initially designed to participate to the Computer Cooking Contest[4] (CCC) challenge, an international scientific challenge that aims the confrontation among systems able to solve cooking problems. Candidate systems must be able to answer queries expressed in natural language by retrieving or adapting existing recipes given the user constraints. The recipe book, common to all participated systems, is a set of textual recipes described by a title, an ingredient list, and a set of preparation instructions. User requests may include constraints on ingredients, dish types and dish origins.

The TAAABLE project involves researchers interested in various knowledge-based systems fields such as case-based reasoning (CBR), ontology engineering, data and text mining, text indexing and hierarchical classification. TAAABLE entered the CCC in 2008 and won the vice-champion award. The ongoing researches on this project aim at improving the efficiency and the possibilities of evolution of the TAAABLE application.

In the next section, we detail how the TAAABLE application addresses the retrieval and the adaptation of recipes.

[4] http://www.wi2.uni-trier.de/eccbr08/index.php?task=ccc

2.1 The TAAABLE application

Fig. 1. TAAABLECBR web user interface.

The web user interface allows the user to enter her query. The system answers a query by returning a set of recipes satisfying the user's query. If adaptations are needed, they rare displayed to the user. Figure 1 shows an example of response for a "pie with orange".

Adapting a recipe consists in replacing some ingredients by some others. By clicking on a given recipe, the user will reach the recipe, including the list of ingredients that have to be substituted. In figure 2, the *Key Lime Pie* recipe is adapted by replacing *key lime juice* and *key lime* by *orange*.

The architecture of the TAAABLE system is composed of two distinct parts. The offline part of the system focuses on the management of the knowledge base and the indexing of the CCC recipes. Specific tools have been developed for that purpose. The result of the offline part is a set of data that will be used for bootstrapping the CBR engine plugged behind the web user interface (in the online part).

The design of these two parts is detailed in the next section.

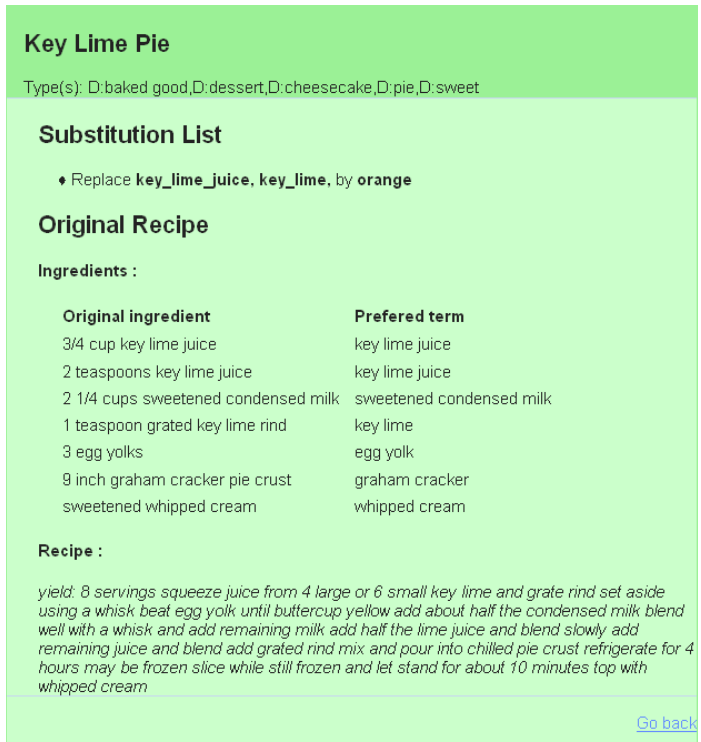

Fig. 2. An example of recipe adaptation.

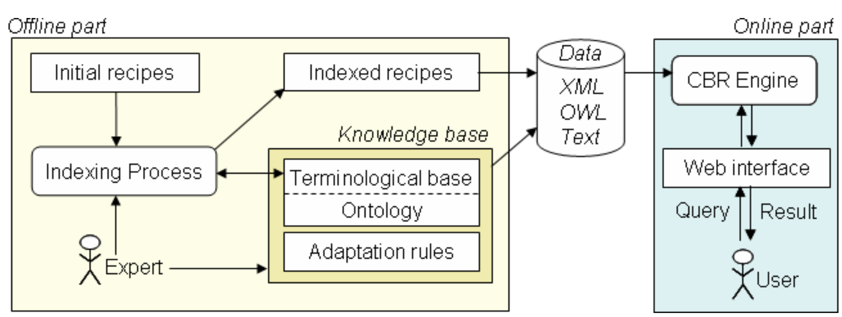

Fig. 3. Architecture of the TAABLE system.

2.2 Knowledge Organization

A hierarchy of ingredients. The CBR engine adapts a recipe by substituting some ingredients by other ones. However, for sake of simplicity, each ingredient involved in the adaptation process of a recipe is replaced by one ingredient (1 to 1 substitution). The underlying idea is that a recipe can be adapted by substituting an ingredient by another "close" ingredient. Therefore, ingredients are organized in an ontology. This ontology is used by the CBR engine to calculate the cost of a substitution: the closer the ingredients, the lower the cost. For instance, *orange* is closer to *lemon* than *apple*.

Hierarchies of recipe types. In order to type recipes, three other hierarchies are defined. *Dish moments* such as appetizer, dessert, . . . , *dish types* such as cake, pizza, . . . and finally *dish origins* such as Mediterranean, Chinese, ...

We call **ontology** in the following, the four above hierarchies. The ontology describes concepts of the domain. The link between the conceptual structure (the ontology) and the linguistic level (recipes) is performed thanks to the **terminological base**: each concept of the ontology is associated with its linguistic forms, *i.e.* a set of term variations. For instance, the concept *litchi* is associated to the terms *litchi*, *lichi*, *lychee*, *leechee*, and *laichee*.

The ontology and the terminological base have been built jointly by the experts. Depending on the results of the indexing process (see below) which may highlight lacks in the terminology, the expert manually decides which terms and concepts (associated to terms, if needed) should be added and the place of the concept in the hierarchy.

In addition to the ontology, the CBR engine may use **adaptation rules** to adapt a recipe. An adaptation rule is an ordered pairs of ingredients sets (s_1, s_2) with a cost given by the expert. (s_1, s_2) stating that the set of ingredients s_1 can be replaced by the set of ingredients s_2.

Adaptation rules and the ontology form the **knowledge base**.

2.3 Indexing the recipes according to the terminological base

The **indexing process** is an automatic process which creates the indexing of the recipes according to the terminological base. It takes as entry recipes in their initial textual form. For each recipe, the output of the indexing process is (1) tagged recipes: an XML (textual) form of the recipe where terms from the terminological base are tagged, (2) a list of concepts indexing recipes written in propositional logic (3) Error reports: a set of ingredient lines in the initial recipe where no term from the terminological base has been identified (possible lack in the base). We detail below these points.

Tagged recipes. The CCC recipes are given in a loosely structured XML format: tags are used for identifying the recipe title (TI element), the ingredients (IN elements), and the preparation (PR element). The indexing process adds tags to the ingredient part of recipes. It searches into the ingredient lines (tagged IN) the presence of terms of the terminological base, and introduces for each term the concept involved. The ingredient line <IN>300 g mashed bananas</IN> will be tagged as <IN><ING>banana</ING><QT>300</QT><U>g</U><QL>mashed</QL><R/> </IN> where <ING>banana</ING> is the concept associated with the term bananas, <QT>300</QT> is the quantity, <U>g</U> is the unit, <QL>mashed</QL> is a “qualifier”, <R/>is the rest of the ingredient line not recognised by the parser.

Here it is empty. The tagged line is used by the expert to control the correctness of the parsing.

Types (dish types, origins, and moments) of a recipe are not explicitly mentionned in the initial form of a recipe. It is automatically computed by three steps process. First, it searches if a recipe with the same title exists in the *recipesource.com* web database with some type information. If it fails, it searches if the title of the recipe contains terms that represents a type of dish or an origine (e.g. *Banana Butterfinger* **cake**). Finally, if step 2 fails, the process uses a set of association rules *ingredient(s)* $\rightarrow$ *type or origin* (e.g. *mascarpone* $\wedge$ *coffee* $\rightarrow$ *tiramisu*) to type the recipe. Association rules have been previously extracted from the complete *recipesource.com* web database.

Type indexation is quite noisy. 30% of recipes are not assigned to any type, some types (moment . . .) are missing, and there are some errors (e.g., "pizza sauce" is not a *pizza*). The experts have to check manually these tags for each recipe.

Indexing recipe in a propositional form. The knowledge representation language used by the CBR engine is a fragment of propositional logic. The ontology is encoded as a set of axioms $a \Rightarrow b$. For example, the axiom `apple` $\Rightarrow$ `fruit` of $\mathcal{O}$ states that any recipe with apples is a recipe with fruits.

All recipes of the recipe book are indexed by a conjunction of literals. For example, the recipe of the apple pie (denoted by R) is indexed by:

$$Idx(R) = \texttt{apple} \wedge \texttt{pie} \wedge \texttt{sugar} \wedge \texttt{pastery}$$

The set of indexed recipes constitutes the case base of the system.

The indexed recipes resulting from the indexing process and the ontology are exploited by the CBR engine to answer user queries.

Error reports. We deal here with two main types of errors. One is coming from bad writing of recipes. The other one is due to concept missing in the ontology. All these errors should be corrected for the CBR engine to run properly.

Bad writing of recipes. Parts of recipes need corrections because of different typographic mistakes:

- `<IN>4 ts Baking power</IN>` should be corrected into `<IN>4 ts Baking powder</IN>`,
- the two consecutive ingredient elements `<IN>1 lb Boneless pork, cut in 3/4</IN><IN>Inch cubes</IN>` should be merge in one ingredient line `<IN>1 lb Boneless pork, cut in 3/4 inch cubes</IN>`,
- `<IN>Salt; pepper, Worcestershire and lemon juice</IN>` should be split into `<IN>Salt</IN><IN>pepper</IN><IN>Worcestershire</IN><IN> lemon juice</IN>`).

Most of these errors have been detected by the experts while checking the tagged recipes.

Missing concepts in the ontology. Some ingredient lines were not indexed by any terms of the terminological base. This means that no ingredient concept was recognised in this line. The expert has to check first that the line is correctly written. If it is correct, the error comes from missing concepts in the ontology or from missing terms in the terminological base. For example, if the `spam` concept [5] does exist in the ingredient hierarchy then the ingredient line `<IN>1/2 cn Spam, in 3/4" cubes</IN>` cannot be indexed. The expert has to add the new concept `spam` in the ontology, defines its position in the hierarchy, and associates this concept to a list of terms (*spam* ...) in the terminological base. Otherwise, if the concept exists in the ontology but the term is not recognised. Then, only the terminological base has to be updated. This error will be corrected at the next run of the indexing process. The same problem exists for the hierarchies of types, moments and origins.

2.4 Case-based reasoning

Querying the system. A request expressed in natural language is processed through the web interface and is formalized in the system by a conjunction of literals. For example, in TAAABLE, the request "I want a dessert with rhubarb but without chocolate" is transformed in a query, denoted Q:

$$Q = \texttt{dessert} \wedge \texttt{rhubarb} \wedge \neg\texttt{chocolate}$$

Retrieval and adaptation mechanism. The first step of the CBR process consists in retrieving among the available recipes, a recipe "similar" to the query. The retrieval is performed by classification on the basis on the query and the recipes indexes. First, a strong classification is applied: the system searches for a recipe whose index matches exactly the index of the query. If strong classification fails, a smooth one is applied: the query is generalized until a satisfactory solution is found [6]. Smooth classification leads to an approximate matching of the results and implies an adaptation of the retrieved recipe for answering the query.

The adaptation of a recipe uses adaptation knowledge. In the first version of TAAABLE, adaptation knowledge is only given by ingredient substitutions. For example, an apple pie recipe can be adapted in a rhubarb pie recipe by substituting apples by rhubarb in the recipe. However, one might like to perform a more complex adaptation of this recipe. This will be possible in a future version of the application. For example, the adaptation of the apple pie recipe will be performed by substituting rhubarb to apples and by adding sugar to the recipe in order to make the recipe less sharp.

2.5 Synthesis

Obviously, the knowledge base used by the CBR engine is neither correct nor complete and needs to be updated and improved. However, due to the independence of the two parts of the TAAABLE system, it is impossible to take into

[5] *Spam* stands for *spiced ham*, a kind of precooked canned meat.

account interactively the user feedback to make evolve the knowledge base of the system. This is a significant limitation of the current architecture. From a more practical point of view, it has also been a limitation during the development of the first version of TAAABLE: we were not able to improve the knowledge base online and any modification was time-consuming.

Hence, for the next version of the application, the goal is to link the two parts, i.e., the CBR engine and the knowledge base management tools, in order to be more efficient and to be able to take into account user feedback in the application.

3 WIKITAAABLE: A semantic wiki for TAAABLE

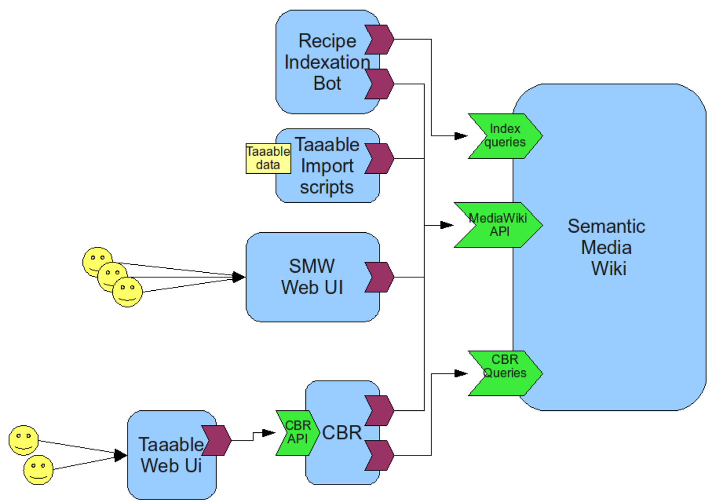

Fig. 4. WIKITAAABLE components.

In this section, we present the new generation of the TAAABLE system, called WIKITAAABLE.[6] In WIKITAAABLE, we address many problems of the TAAABLE application by using Semantic Media Wiki (SMW) [1] as a blackboard. The semantic wiki allows users to browse, query, edit, and validate the knowledge base as pointed out in [7]. In addition, the knowledge base can be updated by results produced by the CBR engine and by the automatic indexing tool. The architectural view of the system is presented in figure 4.

[6] WIKITAAABLE is not yet available for public because of the Computer Cooking Contest.

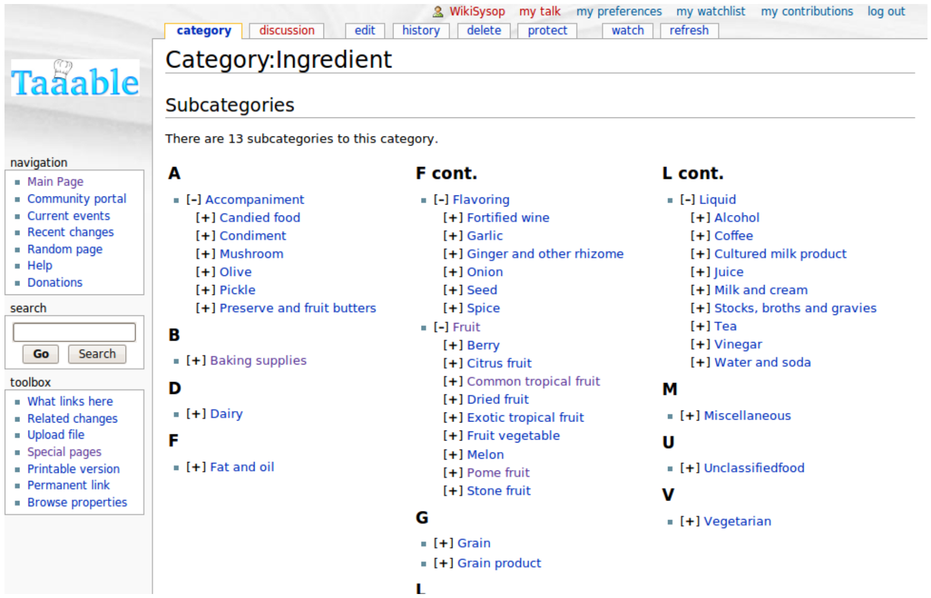

Fig. 5. WIKITAAABLE ingredient ontology.

Semantic Media Wiki. Semantic Media Wiki is used as a blackboard by users, the CBR engine and the recipe indexing bot. The CBR engine and the recipe indexing bot rely on a set of predefined semantic queries to gather their inputs. The knowledge base is represented by a graph of semantic wiki pages. We have represented as semantic wiki pages the indexed recipes, the ingredient ontology and the ontology of dish types (see figure 5).

Mediawiki Web User Interface. This is the regular user interface of semantic media. Through this interface, users can add new recipes and modify the ingredients, the types of dishes and the origins of dishes.

Recipe Indexing Bot. The recipe indexing bot crawls the recipe pages, extracts ingredient information, and updates recipe pages with semantic indexings and categorization of recipes. The crawling and updates of recipes is done using the mediawiki API, accessing the knowledge base is done using predefined semantic queries. The following example illustrates the input of the Recipe Indexing Bot:

```
== Ingredients ==
*   1 c rice
*   2 c water
*   1/2 c sugar
*   1 ts salt
*   2 c evaporated milk
*   1 c raisins
*   3 eggs separated
*   3/4 ts vanilla
```

Fig. 6. indexed recipe of "ARROZ DULCE".

```
*   1/4 ts cinnamon
*   1/4 ts nutmeg

== Preparation ==

* combine the rice water sugar and salt in a large saucepan bring the
 water to a boil and cover the saucepan reduce the heat to low and
 continue to cook for 12 - 15 minutes or until the water is absorbed
 combine the milk and egg yolk add them to the rice then mix in the
 raisin vanilla and cinnamon simmer for five minutes remove from the
 heat beat the egg white until stiff fold them into the rice chill and
 garnish with nutmeg before serving also taste good warm
```

The above recipe is indexed in the semantic wiki as presented in figure 6. We used the n-ary relationship of Semantic Media Wiki to represent an ingredient line.

CBR. The CBR engine retrieves its knowledge base through predefined semantic queries. Next, it is able to answer requests issued using the CBR web interface

Fig. 7. TAAABLE user feedback.

(cf. figure 1). Our idea is that the CBR user interface proposes recipes based on adaptation of existing recipes. Users are invited to give feedback about recipes. The figure 7 illustrates how we can capture the feedback of the user. This interface is currently not functional.

We imagine that the WIKITAAABLE will work like this: Suppose a user requests "pie" and "orange" thinking about an "orange pie". The system presents results as in figure 7. The user chooses "I don't like it" for the first proposal because replacing lemon juice by orange in an "apple crumble pie" does not transform an "apple crumble pie" into an "orange Pie". However, the proposal 2 and 3 are acceptable. So the user select "I like it" for these two proposals.

If the feedback is positive, then the new recipe is added to the knowledge base and a new semantic wiki page is created for it. The new recipe is marked as "generated". If the user feedback is negative, then the computed recipe is also added to the knowledge base and a new page is created. However,this new page belongs to the category "refused". These recipes are kept for future reuse. For instance, experts can analysis refused recipes and use them as a basis for a failure driven knowledge acquisition [8].

TAAABLE Import Scripts. We have written scripts to import the current knowledge base of TAAABLE into Semantic Media Wiki. The recipe base contains about 888 recipes, the ingredient ontology contains 8506 different classes of ingredients. We used *RAP - RDF API for PHP* to parse the ontology.

4 Discussion and future work

The strength of the semantic wiki comes from the facility for the community to update and enrich a set of annotated recipes as well as the ontology. The WIKITAAABLE system has many advantages compared to the original one:

- Users can add new recipes.
- Users can correct indexation of recipes.
- Users can browse and navigate through the ontology.
- Ontology maintainers can quickly modify the ingredient ontology or the dish type ontology and test the effects on the CBR engine.
- The feedback about adaptation of recipes can now be captured and represented in the semantic wikis. The knowledge base of the system increases just by using it.

One of the reasons of the success of CBR systems is that they are theoretically able to "learn from experience" by acquiring additional knowledge with each problem solving session. In TAAABLE, however, learning from experience is difficult because of the lack of an embedded mechanism allowing to use feedback for improving existing knowledge bases. A strength of the use of Semantic Media Wiki in WIKITAAABLE is that it facilitates this process by enabling the enrichment and the update of data and knowledge by a community of users. The management of the ontology and the annotated recipes have many advantages compared to this of the previous version of the application. Indeed, users can add new recipes, correct indexing of recipes, and browse and navigate through ontologies. Ontology maintainers can as well easily modify the ontologies and test the impact of the modifications on the results of the CBR engine. Besides, a major advantage is that the feedback on the adaptations made by the CBR engine can be captured and represented in the wiki.

However, the development of WIKITAAABLE raises several issues that are mainly related to the coherence of the system. How can we guarantee the coherence of the systems while several users, often having different viewpoints, are allowed to modify the knowledge coded in the system? How can we efficiently combine a semi-automatic procedure and a manual enrichment process to build an ontology? These issues are of major importance because the ontology plays a central role in the TAAABLE system and is mainly used at two different levels. It guides the indexing process by identifying concepts involved in each recipe, and it is used by the CBR system to adapt recipes. If any user can freely modify the ontology, then the CBR engine and the recipe indexing bot might produce unpredictable results. Several strategies can be envisioned to tackle this problem:

– Restrict the update of ontologies to "administrators". This is a limitation to the collaborative work. Moreover, the improvement of the ontology strongly depends on the availability of the administrators.
– Validation of changes before they are integrated in the running system. This strategy has also several limitations: poor process support by semantic wikis, synchronization problems between several versions of a single system, and time consuming for "administrators". Moreover, this does not solve the problem of conflicts between several concurrent changes.
– Adaptation of continuous integration approaches used in software engineering in the context of the WIKITAAABLE system. For example, if an adaptation of a recipe has been validated by several users, then the ontology should be modified in order to preserve this adaptation (in this case, user feedback collected by WIKITAAABLE should generate non-regression tests).
– Toward a peer-to-peer WIKITAAABLE? In such an approach, each user would have his/her own version of the application, relying on a common knowledge base, and would be able to perform personal adaptation of the knowledge. Adaptation would be shared between users on the basis on the confidence they have in their peers.

The next step of the development of WIKITAAABLE is to fully integrate the CBR engine to the Semantic Wiki and to set up interaction possibilities at several levels. One particular focus will be put on the ability of the semantic wiki to represent and manage complex adaptation rules, including Boolean constraints.

References

1. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic wikipedia. Journal of Web Semantics **5**(4) (2007)
2. Schaffert, S.: IkeWiki: A Semantic Wiki for Collaborative Knowledge Management. 1st International Workshop on Semantic Technologies in Collaborative Applications (STICA06), Manchester, UK (2006)
3. Buffa, M., Gandon, F.L., Ereteo, G., Sander, P., Faron, C.: Sweetwiki: A semantic wiki. Journal of Web Semantics **6**(1) (2008) 84–97
4. Badra, F., Bendaoud, R., Bentebitel, R., Champin, P.A., Cojan, J., Cordier, A., Desprès, S., Jean-Daubias, S., Lieber, J., Meilender, T., Mille, A., Nauer, E., Napoli, A., Toussaint, Y.: TAAABLE: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. In Schaaf, M., ed.: ECCBR 2008, The 9th European Conference on Case-Based Reasoning, Trier, Germany, September 1-4, 2008, Workshop Proceedings. (2008) 219–228
5. Riesbeck, C.K., Schank, R.C.: Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey (1989)
6. Lieber, J.: Strong, Fuzzy and Smooth Hierarchical Classification for Case-Based Problem Solving. In van Harmelen, F., ed.: Proceedings of the 15th European Conference on Artificial Intelligence (ECAI-02), Lyon, France, IOS Press, Amsterdam (2002) 81–85

7. Krötzsch, M., Schaffert, S., Vrandecic, D.: Reasoning in semantic wikis. In Antoniou, G., Aßmann, U., Baroglio, C., Decker, S., Henze, N., Patranjan, P.L., Tolksdorf, R., eds.: Reasoning Web. Volume 4636 of Lecture Notes in Computer Science., Springer (2007) 310–329
8. Cordier, A.: Interactive and Opportunistic Knowledge Acquisition in Case-Based Reasoning. PhD Thesis, Université Claude Bernard Lyon 1 (2008)